A Blockchain-Based Optimized and Secure Scheduling Framework for Charging Service

Journal:	*Transactions on Services Computing*
Manuscript ID	TSC-2024-08-0724
Manuscript Type:	Regular Paper (S1)
Keywords:	Electric vehicle, charging scheduling, smart contract, blockchain

A Blockchain-Based Optimized and Secure Scheduling Framework for Charging Service

Xinyu Lu, *Graduate Student Member, IEEE*, Zihan Wang, *Graduate Student Member, IEEE*, Zhuo Chen, *Student Member, IEEE*, Jiong Lou, *Member, IEEE*, Chentao Wu, *Member, IEEE*, Yuan Luo, *Member, IEEE*, Guangtao Xue, *Member, IEEE*, Yusheng Ji, *Fellow, IEEE*, Wei Zhao, *Fellow, IEEE*, Jie Li, *Fellow, IEEE*

Abstract—With the global focus on reducing carbon emissions, the proliferation of electric vehicles has significantly increased, catalyzing the development of corresponding charging infrastructure. Despite these advancements, substantial challenges remain in aligning the availability of charging facilities with the rising charging demand. Numerous cloud-based and blockchain-based platforms have been introduced, yet these solutions raise concerns regarding optimality, security, fairness, and efficiency. This paper introduces a novel electric vehicle scheduling framework based on blockchain technology, which unifies the supply and demand sides of the charging system through an inclusive formulation. The proposed smart contract-based scheduling algorithm employs a threshold-based approach to capture the supply and demand dynamics, ensuring theoretical guarantees and fair resource allocation. Furthermore, this paper introduces an innovative node selection algorithm that considers the characteristics of different charging providers to enhance blockchain consensus speeds. Rigorous theoretical proofs are provided to demonstrate the guarantees of the proposed scheduling algorithm. Security analysis and extensive experiments have been conducted to validate the framework's security and efficiency.

Index Terms—Electric vehicle, charging scheduling, blockchain, smart contract

1 Introduction

Given the global emphasis on reducing carbon emissions and promoting clean energy, the proliferation of electric vehicles (EVs) has reached a significant scale [1]. This trend not only fosters the transformation and upgrading of the automotive industry but also imposes higher demands on the associated supporting infrastructure. Consequently, the development of charging piles, charging stations, and other related facilities is accelerating to meet the increasing demand of EV charging [2]. Nevertheless, the widespread adoption of electric vehicles presents additional challenges due to the inherent difficulties in effectively aligning the availability of charging facilities with the charging demand.

The core issue of the charging system lies in the imbalance between supply and demand. Geographically, the distribution of charging stations is uneven, with dense coverage in economically developed areas and sparse coverage in remote and rural regions. In recent years, researchers have increasingly explored potential charging resources, including the consideration of private charging piles as a service [3], the utilization of dedicated charging piles for service provision [4], and the implementation of mobile charging piles as a service [5]. These approaches have, to some extent, enhanced the coverage of charging resources. Temporally, there exists an imbalance between charging requests and the availability of charging facilities. The heterogeneous nature of these facilities results in varied utilization rates and unpredictable availability times of charging piles. For instance, a private charging pile is accessible only when it is not in use by its owner, and dedicated charging resources become available only upon the completion of mandatory tasks. Even within public charging stations, where resources are presumed to be consistently available during fixed hours, the highly dynamic nature of charging requests poses significant challenges to the scheduling of limited resources.

To address supply and demand matching, various cloud-based platforms have been proposed commercially [6], [7]. Electric vehicle owners access these platforms via mobile data networks to submit their charging requests. The server

- *Xinyu Lu, Zihan Wang, Jiong Lou* (Corresponding author) *and Yuan Luo are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: {gale13, Zihan.Wang, lj1994, yuanluo}@sjtu.edu.cn.*
- *Jie Li* (Corresponding author) *and Chentao Wu are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, with the Yancheng Blockchain Research Institute, and with the Shanghai Key Laboratory of Trusted Data Circulation and Governance in Web3, China. E-mail: lijiecs@sjtu.edu.cn;wuct@cs.sjtu.edu.cn.*
- *Zhuo Chen is with the Department of Computer Science, Bryn Mawr College, PA 19010, USA. E-mail: zchen4@brynmawr.edu.*
- *Guangtao Xue is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, and with the Shanghai Key Laboratory of Trusted Data Circulation and Governance in Web3, China. E-mail: gt_xue@sjtu.edu.cn.*
- *Yusheng Ji is with the Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo 101-8430, Japan. E-mail: kei@nii.ac.jp.*
- *Wei Zhao is with the CAS Shenzhen Institute of Advanced Technology, Shenzhen 518055, China. E-mail: zhao.wei@siat.ac.cn.*
- *This work has been partially supported by the National Key R&D Program of China No. 2020YFB1710900 and No. 2020YFB1806700, NSFC Grants 61932014 and 62232011.*

then employs a recommendation algorithm—or allows the electric vehicle driver to directly specify preferences—to allocate available charging resources accordingly. However, this framework presents three primary challenges. First, the optimality of the recommendation results provided by the server remains uncertain, lacking theoretical or practical guarantees. Second, a centralized system introduces potential safety hazards. The system operator must be entirely trustworthy; otherwise, user utility may be compromised. Third, the server's strategy may lack transparency, potentially resulting in favoritism towards certain charging facility providers and thereby raising fairness concerns.

Considering the optimality of charging service recommendations, as well as the security and fairness of the platform, previous research can be broadly categorized into two distinct approaches: centralized scheduling optimization [4], [13], [14] and decentralized scheduling based on secure frameworks [8]–[10]. Blockchain (BC), as a significant distributed technology, has been extensively studied [3]. However, optimization-based solutions often overlook the role of intermediary platforms, while scheduling embedded within secure blockchain platforms typically employs first-come-first-served heuristics or greedy approaches, thereby rendering their optimality uncertain. Consequently, these methods fail to align electric vehicle charging demand and supply effectively in real-world scenarios. Achieving a secure platform alongside optimization remains a significant challenge. A detailed comparison of recent works is summarized in Table 1.

To comprehensively address the aforementioned challenges, this paper introduces an electric vehicle scheduling framework grounded in blockchain technology. This framework unifies the supply and demand sides of the charging system through a single formulation that encompasses various types of charging resources and employs a smart contract-based scheduling algorithm to facilitate the automatic matching of supply and demand. Notably, the proposed scheduling algorithm utilizes a threshold-based approach to capture the spatiotemporal dynamic features of the supply and demand sides. This approach ensures theoretical guarantees and enables the algorithm to operate within smart contracts in a distributed manner, thereby enhancing the fairness of the scheduling mechanism. Additionally, we propose a novel node selection algorithm that considers the characteristics of different charging service providers, ultimately contributing to improved efficiency and mitigating the risk of centralization on the underlying blockchain infrastructure.

The contributions of this work are summarized as follows:

- **Formulation and Optimization**: We undertake a comprehensive modeling of the structure and attributes of the EV charging system and introduce a novel threshold-based charging scheduling algorithm. This algorithm functions entirely in an online manner, independent of any distributional assumptions or future information, and is substantiated by specific theoretical guarantees.
- **Fairness and Security**: We propose a blockchain-based distributed EV charging scheduling framework that effectively balances supply and demand. Our design employs a scheduling process based on smart contracts to ensure participant fairness. The decentralized nature of this framework mitigates security vulnerabilities.
- **Lightweight and Efficient**: We present an innovative node selection algorithm designed to significantly reduce consensus computations within blockchain networks, thereby improving overall system efficiency. The algorithm's lightweight nature further guarantees the sustained efficiency of the system.

We demonstrate the effectiveness of the proposed framework through rigorous theoretical proofs, comprehensive security analyses, and experimental validation using real-world datasets.

2 Related Works

A significant body of research has been devoted to balancing charging demand with available charging resources. These efforts can be broadly classified into two distinct categories. The first category presumes the existence of a cloud-based platform that optimizes supply and demand matching and scheduling using aggregated data. The second category prioritizes the design of secure frameworks, often leveraging the decentralized architecture of blockchain technology to meet the procedural demands of charging scheduling.

2.1 Optimized EV Charging Scheduling

Researchers optimize charging scheduling by incorporating a variety of decision-making perspectives and behavioral styles. A significant approach involves analyzing the dynamic response of charging behavior to price fluctuations and employing pricing strategies to influence demand-side conditions. For instance, considering time-variant and heterogeneous demand, Yating Ding *et al.* [15] propose a consumer-perspective utility model to assist charging service providers in optimizing pricing decisions. From the perspective of boosting charging network operators' profits and avoiding excessive charging costs for electric vehicles, some researchers [16] have proposed a three-level Stackelberg game involving the distribution network operator and users. In other studies, researchers [4], [13] formulate the operation of a charging station as a queuing system, proposing an optimal pricing strategy to guide and coordinate the EV charging processes within the station. With advancements in machine learning, an increasing amount of research is employing reinforcement learning and other methodologies to deliver intelligent pricing services [14], [17].

The previously described method of utilizing economic incentives to regulate the load on the charging system is considered a 'soft' approach. In contrast, a substantial body of research focuses on more direct charging scheduling strategies. Some scholars address the issue from a geographical perspective, optimizing the distribution of public EV chargers or charging stations within urban areas [18], [19]. Others [20] concentrate on EV scheduling by solving a mixed linear programming problem for the joint routing and charging schedule of EVs. Weijia Zhang *et al.* [21] propose a comprehensive Multi-Agent Spatio-Temporal Reinforcement Learning framework that intelligently recommends charging stations by considering factors such as charging waiting time and charging failure rate. In scenarios with

TABLE 1: Comparative Analysis of Existing Mechanisms

Mechanism	Targeted Task	Optimization		Security	Efficiency	
		Technique	Theoretical Guarantee	Technique	Complexity[a]	Enabled in BC
[8]	Efficient and Fair Trading	Heuristic Process	✗	Blockchain	$\mathcal{O}(\Pi\|\mathcal{SP}\|)$	✓
[3]	Secure and Cooperative Sharing	Matching-Coalition Game	✓	Blockchain	$\mathcal{O}\left(n\|\mathcal{SP}\|^2\right)$	✗
[9]	Energy Exchange	-	✗	Blockchain, DL	-	-
[10]	Secure and Optimized Trading	Improved KH Algorithm	✓	Blockchain	High (iterations)	✗
[11]	Privacy and Safety Aware Control	Reinforcement Learning	✗	Differential Privacy	High (training process)	-
[4]	Real-time Scheduling	Lyapunov Optimization	✓	-	$\mathcal{O}(\|\mathcal{SP}\|(N+n\log n))$	✗
[12]	Efficient and Private Scheduling	Reinforcement Learning	✗	Extra Battery	High (training process)	-
Proposed	Optimized and Secure Scheduling	Threshold-based	✓	Blockchain	$\mathcal{O}(\overline{D}\|\mathcal{SP}\|)$	✓

[a]The complexity discussed in the original paper is adapted to suit the scenario presented in this study. The following notations are utilized, inspired by related works: Π, representing the number of parameters considered in [8]; $|\mathcal{SP}|$, denoting the number of charging service providers; N, indicating the number of charging piles; and n, the number of EVs. $\overline{D}$ is the maximum charging duration.

stringent time constraints, researchers have also investigated charging behavior scheduling to ensure adherence to schedules [22], [23].

This category of research primarily focuses on the optimality of scheduling, formulating optimization problems using various methodologies and considering a range of influencing factors, ultimately leading to enhanced scheduling decisions. However, such designs often overlook the role of middleware [21] or assume the existence of a cloud-based platform [4] that connects charging stations with electric vehicles. This assumption may introduce security concerns, such as the single point of failure of the platform and the potential leakage of private information related to vehicles.

2.2 Secure Charging Scheduling

Given the aforementioned challenges, a multitude of studies have harnessed blockchain technology to establish a distributed and secure framework for electric vehicle scheduling. For example, Riya Kakkar *et al.* [24] developed a decentralized application based on the Ethereum blockchain for EV charging at charging stations and facilitating energy trading between prosumer and consumer EVs. Similarly, Yuntao Wang *et al.* [3] proposed a blockchain-based secure private charging pile (PCP) sharing scheme, which models the interactions among electric vehicles and cooperative PCPs as a joint coalition-matching game. Additionally, other researchers [25] have introduced a privacy-preserving and traceable blockchain-based charging payment (PTB-CP) scheme for EVs.

In a broader context, numerous studies have focused on the development of blockchain-based Energy Internet of Things (EIoT) systems. For instance, Jindal *et al.* [26] proposed a blockchain-based secure demand response management scheme, aimed at facilitating secure energy trading decisions and managing the overall load across residential, commercial, and industrial sectors. Similarly, Li *et al.* [27] introduced a blockchain-based intelligent and fair Internet of Vehicles (IoV) charging service system. Furthermore, Sun *et al.* [8] proposed an efficient and secure trading framework utilizing multiple consortium blockchains, incorporating a fair and robust trading strategy implemented through smart contracts. In addition to blockchain technology, deep learning (DL) and ϵ-differential privacy have been introduced to further enhance security [9], [11].

In this type of work, the scheduling process is typically treated as a heuristic procedure [8], or the scheduling algorithm is decoupled from the blockchain system [3], [10]. The blockchain system either lacks guarantees based on optimization theory or suffers from inefficiencies in practical implementation. Therefore, the problem addressed in this paper is nontrivial, as it necessitates the simultaneous resolution of both optimization and security characteristics of the system, along with enhancements in its efficiency.

3 System Model

In this section, we provide a conceptual introduction to the fundamental operational processes of the charging system, as well as the overarching optimization objectives. In section 4, we will delineate the detailed implementation of these abstract concepts within the blockchain-based system.

3.1 Overview

As shown in Fig. 1, an urban EV charging system comprises three primary entities: the supply side, the demand side, and the middleware. The supply side encompasses various charging service providers, including public charging stations, dedicated charging stations, private charging piles, and mobile charging equipment. Assuming there are K providers in the system, the set of providers can be denoted as $\mathcal{SP} = \{1, 2, \cdots, k, \cdots, K\}$. The demand side comprises electric vehicles within the city, denoted as $\mathcal{EV} = \{1, 2, \cdots, i, \cdots, I\}$. A middleware is essential to facilitate matching and trading between supply and demand entities, providing matching recommendation results and executing specific transactions. Typically, this middleware is either a centralized cloud-based platform or a trust implemented by a distributed approach, such as the blockchain in this paper.

3.2 Charging Supply and Demand

To enhance operational precision, the timeline of the charging system is partitioned into equal-sized time slots $\mathcal{T} = \{1, 2, \cdots, t, \cdots, T\}$. Fully considering the fluctuations in the number of available charging piles at charging stations and the dynamic nature of the available time of private charging piles, we adopt a unified time resource occupancy model to assess the availability of various charging piles. This approach involves measuring resources in terms of

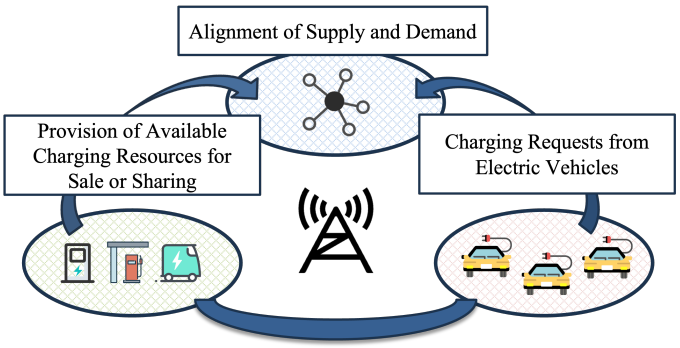


Fig. 1: Charging System

time rather than the charging power. Based on this, we define the status of the charging service provider as follows.

Definition 1 (Charging Supply Status)**.** *For the kth charging service provider, its status can be denoted as* $Q_k = \{g_k, p_k, \boldsymbol{c}_k, N_k, \boldsymbol{A}_k\}$*. Here,* g_k *represents the geographical location of the charging provider, which may be a set in the case of mobile charging service providers.* p_k *denotes the charging power of the charging station, and* $\boldsymbol{c}_k$ *is a* T*-dimensional vector representing the charging price at each time slot.* N_k *represents the number of charging piles owned by the charging provider.* $\boldsymbol{A}_k$ *is a* T*-dimensional vector indicating the availability of the charging service provider, where* $\boldsymbol{A}_k(t) \in [N_k] \cup \{0\}$ *and* $[N_k] = \{1, 2, \cdots, N_k\}$*. If* $\boldsymbol{A}_k(t) > 0$*, it signifies that the charging pile is available at time* t*; otherwise, the charging pile is unavailable.*

Upon reviewing the pricing, location, and other relevant details of the charging service provider, EVs are able to submit a charging request to the middleware.

Definition 2 (Charging Request)**.** *The middleware receives a sequence of charging requests* $\mathcal{CR} = \{1, 2, \cdots, j \cdots, J\}$ *over the entire duration. Each charging request* R_j *follows the format* $R_j = \{i, g_j, a_j, E_j, s_j, \overline{d}_j\}$*, where* i *denotes the EV submitting the charging request;* g_j *represents the location of the corresponding EV;* a_j *indicates the arrival time of the charging request, and* s_j *represents the anticipated charging start time. Additionally,* E_j *(kWh) is the requested amount of energy and* $\overline{d}_j$ *denotes the maximum acceptable charging duration for EV* i*.*

- *Upon receiving the charging request, the middleware conducts a detailed analysis. Initially, it filters out charging service providers located beyond a certain distance from the requesting EV's geographical location, forming an optional set denoted as* $O_j = \{k \mid dis(g_k, g_j) < \zeta\}$*. Subsequently, to standardize the format, the charging requests can be structured, detailing the specifics for each charging service provider. For provider* k*, the charging information can be represented as* $R_j^k = \{v_j^k, T_j^k\}$*, where* $v_j^k = f(\boldsymbol{c}_k)$ *is a function associated with the charging price, representing the utility of the charging request. Specifically, from the provider's standpoint,* v_j^k *is positively correlated with the price.* $T_j^k = \{s_j, s_j + 1, \cdots, s_j + d_j^k - 1\}$ *comprises* $d_j^k = \lceil \frac{E_j}{p_k} \rceil$ *consecutive time slots, denoting the time resources required for the charging request. If* $d_j^k > \overline{d}_j$*, provider* k *will be excluded from the optional set* O_j*.*

3.3 Overall Objective

According to *Definitions* 1-2, we formally define the Offline Utility Maximization problem as ***oum***.

$$\max_{\boldsymbol{x}} \quad \sum_{j \in \mathcal{CR}} \sum_{k \in \mathcal{SP}} v_j^k x_j^k \tag{1}$$

$$s.t. \quad \sum_{k \in O_j} x_j^k \leq 1, \ \forall\, j \in \mathcal{CR}, \tag{2}$$

$$\sum_{k \in \mathcal{SP} \setminus O_j} x_j^k = 0, \ \forall\, j \in \mathcal{CR}, \tag{3}$$

$$\sum_{j \in \mathcal{CR} | t \in T_j^k} x_j^k \leq \boldsymbol{A}_k(t), \ \forall\, t \in \mathcal{T}, \forall\, k \in \mathcal{SP}, \tag{4}$$

$$x_j^k \in \{0, 1\}, \ \forall\, j \in \mathcal{CR}, \ \forall\, k \in \mathcal{SP}. \tag{5}$$

Optimization Objective: The objective of problem ***oum***, as delineated in (1), aims to optimize the utility of the entire charging system. This overarching objective is obtained by aggregating the contributions from all charging requests and charging service providers.
Decision Variables: The decision variables for the charging system are denoted as $\boldsymbol{x} = \{x_j^k\}$. In this representation, x_j^k indicates the acceptance of charging request j, with $x_j^k = 1$ indicating that charging request j is assigned to provider k.
Constraints: Constraint (2) specifies that each charging request should be allocated to at most one charging service provider. Constraint (3) stipulates that charging requests may not be allocated to charging service providers that are absent from the optional set. Constraint (4) ensures that, within each time slot, a charging service provider can serve at most $\boldsymbol{A}_k(t)$ charging requests. Constraint (5) reflects the domain of decision variables $\boldsymbol{x}$.

4 Mechanism Design

It is crucial to recognize that while the structure of the ***oum*** problem aligns with the conventional offline 0-1 integer programming model, the key parameters within this problem, such as $\boldsymbol{A}_k(t)$ and $\mathcal{CR}$, are not predetermined but instead emerge progressively over time. *(i) From a platform perspective*, this necessitates the establishment of a middleware capable of gathering this information in real time and utilizing the collected data to facilitate prompt scheduling decisions. *(ii) From an algorithmic perspective*, the dynamic arrival of charging requests necessitates the development of sophisticated scheduling strategies to accommodate the high demands of real-time processing. In this section, we initially discuss the blockchain-based scheduling management backbone, followed by an introduction to an efficient, performance-guaranteed lightweight scheduling algorithm.

4.1 Blockchain-based Framework

Given the competitive dynamics and mutual distrust among charging service providers within the charging system, we advocate for the adoption of decentralized technologies, specifically blockchain, to guarantee the security and fairness of the charging scheduling management process.

Blockchain can be conceptualized as a sequentially organized structure of blocks, where each block constitutes an ordered list of transactions (Tx). Users engage with the blockchain system by creating an on-chain account,

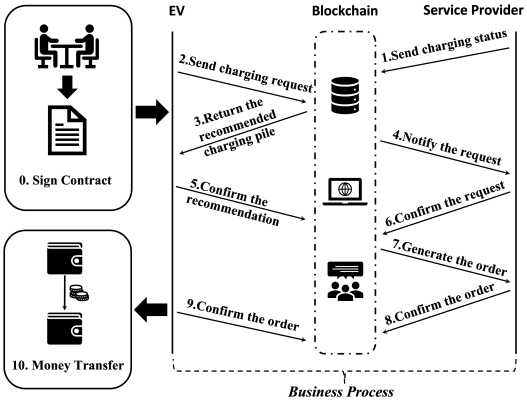


Fig. 2: Transaction Process.

which they utilize to initiate transactions for updating the blockchain's state. Smart contracts are conceptualized as a suite of functions deployed on a blockchain, enhancing its capabilities beyond mere peer-to-peer transactions. These contracts, essentially computer programs residing on the blockchain, operate based on "if...then..." (IFTTT) logic, ensuring execution in strict accordance with the predefined code. By invoking specific functions within a smart contract, transactions can be conducted on the blockchain to fulfill a variety of objectives.

Transactions, generated by user accounts, include a data field equipped with a cryptographic signature and adhere to a specific structure. In blockchains that support smart contracts, transactions can be classified into various types: regular transactions, which involve the transfer of assets from one account to another; contract deployment transactions, which entail the deployment of prewritten code logic to the blockchain, with the contract code embedded in the data field; and contract execution transactions, which facilitate interactions with deployed smart contracts.

The comprehensive process of information flow and transaction generation in the proposed framework is delineated in Fig. 2. The process encompasses some key stages. Initially, the charging service providers *sign a contract*, which involves collaborative negotiations with the EV owner to establish the contents of the smart contract and employs a programming language to construct it. Upon the completion of this definition phase, the smart contract file associated with the deployment contract is disseminated on the blockchain. Next, in the *business process* stage, the charging service provider and EV can invoke functions within the smart contract to adjust their resource availability or submit charging requests. Finally, upon mutual confirmation of the successful conclusion of the business process, the *money transfer* event is initiated.

In a blockchain system, the process of confirming a specific transaction involves the following steps: (i) The trader constructs and publishes a transaction, with the specific functions being executed by invoking a smart contract deployed on the blockchain. (ii) The transaction is broadcast to miners, who then verify and achieve consensus. (iii) Once a miner successfully mines a block that includes the transaction, and the block is subsequently added to the blockchain, the transaction within that block is confirmed.

TABLE 2: Aavailable Energy and Computational Resources

Charging Service Provider	Feature Description	
	Energy	*Computation*
Public charging station	Charging capacity of all charging piles	#CPU/GPU
Dedicated charging station	Capacity of remaining charging piles	#CPU/GPU
Private charging pile	Available time of the charging pile	#CPU/GPU
Mobile charger	Average hotspots access frequency	#CPU/GPU

On a permissionless blockchain supporting Turing-complete languages, smart contracts are employed to implement the principal functionality of the scheduling algorithm. To utilize the scheduling function, users need only publish transactions to invoke specific functions within the smart contract. The smart contract then autonomously calculates the optimal charging service provider based on the predefined algorithmic logic. This autonomous scheduling mechanism not only augments the operational efficiency of the charging network but also bolsters the system's security and fairness.

4.2 Miner Node Selection

Participants in blockchain-based charging systems can be categorized into two groups: ordinary users and miner nodes. *Ordinary users* submit business requests and conduct transfer transactions in a manner comparable to using traditional applications by generating externally owned accounts (EOA) or contract accounts. Electric vehicles, as participants in the blockchain-based charging system, leverage the distributed nature and cryptographic primitives of blockchain technology to achieve secure and transparent transactions. *Miner nodes* deploy substantial computing and storage resources to process, verify, and package transactions, working collaboratively to sustain a globally consistent blockchain ledger. Charging service providers function as miner nodes, upholding the system's security while delivering charging services.

Proof-of-Work (PoW) is a widely recognized consensus algorithm, albeit notably computation-intensive. To address efficiency in the verification process, Proof-of-Stake (PoS) is employed as a more efficient alternative. However, the PoS-based miner node selection algorithm is fraught with issues such as centralization risks and wealth concentration. To enhance consensus efficiency while ensuring fairness among service providers within the charging system, we propose an innovative node selection algorithm designed to optimize the verification process.

Considering the distinct characteristics of various charging service providers within the charging system, we extract availability features of *energy* and *computational* resources corresponding to different types of providers (Table 2). The symbol "#" is used in the table to denote "the number of." Specifically, we utilize the total charging capacity of public charging stations as a metric to assess their energy availability. For dedicated charging stations, given that their charging resources are not consistently accessible to the public [4], we consider only those capacities that are openly available. The operational time of private charging piles is variable; therefore, these time slots should serve as indicators of their availability. For mobile chargers, which may relocate to high-demand areas during holidays to offer charging services [28], the frequency of their visits to

Algorithm 1: Miner Node Selection (MNS)

Input: $\mathcal{SP}, \{\tau_\kappa\}_{\forall\kappa}$
Output: $miner$
$S_c \leftarrow \emptyset$;
// Filtering Phase;
for $\kappa \leftarrow$ {*public charging station, dedicated charging station, private charging pile, mobile charger*} **do**
  for $k \leftarrow \mathcal{SP}$ *and* $type(k) = \kappa$ **do**
    Get the energy and computational resource availability $E(k)$ and $C(k)$ of k;
    Calculate $score(k)$ according to (6);
    if $score(k) > \tau_\kappa$ **then**
      $S_c \leftarrow S_c \cup \{k\}$;
    end
  end
end
// Selection Phase;
$miner \leftarrow$ random select a node in S_c;
return $miner$.

these hotspots is employed to gauge their availability. Regarding the availability of computing resources within the blockchain framework, it is imperative that miners possess the capability to operate a server for executing fundamental computational tasks (storing blockchain state copies and performing transaction verification tasks). Consequently, we quantify the computing resources of a specific charging service provider by assessing the number of computational resources available in their server.

For charging service provider k, the availability of energy resources is denoted as $E(k)$, and the availability of computing resources is represented as $C(k)$. The category to which a charging service provider belongs is designated as $type(k)$. Consequently, the weighted score for each charging service provider can be calculated as follows:

$$\begin{cases} f_k^e = \frac{E(k)}{\sum_{k'|type(k')=\kappa} E(k')} \\ f_k^c = \frac{C(k)}{\sum_{k'|type(k')=\kappa} C(k')} \\ score(k) = f_k^e \times \lambda_1 + f_k^c \times \lambda_2 \\ \text{s.t. } \lambda_1 + \lambda_2 = 1 \quad \lambda_1 > 0, \lambda_2 > 0 \end{cases} . \tag{6}$$

Here, $\kappa \in$ {public charging station, dedicated charging station, private charging pile, mobile charger} and $type(k) = \kappa$. Subsequently, for each type of charging service provider, a specific threshold, denoted as τ_κ, is established. If the score of a charging service provider, represented as $score(k)$, exceeds τ_κ, then that provider is included in the set of miner candidates S_c. Subsequently, a random selection of a node is drawn from these candidates. The algorithm is summarized in Algorithm 1.

4.3 Smart Contract-Based Charging Scheduling

The blockchain's network architecture, along with the participation of miners, establishes a robust foundation for the scheduling mechanism within the charging system. Concurrently, to maintain the integrity and fairness of this scheduling process, it is imperative that the scheduling algorithm be implemented on the blockchain via smart contracts. Despite the Turing completeness attributed to smart contracts, considerations of performance and the practicalities of deployment necessitate that these contracts remain minimally complex. To this end, we have streamlined the intricate processes of information gathering and scheduling into three distinct functions, wherein the scheduling function requires only three straightforward checks, ensuring clarity and simplicity in its logic (Fig. 3). In the following, we will provide a detailed introduction to each of these functions.

`Submit Charging Status()`: In *Definition* 1, the quantities $\boldsymbol{c}_k$ and $\boldsymbol{A}_k$ in the *Charging Supply Status* are not predetermined but are values that can be observed or ascertained at time t. Consequently, a singular *Charging Supply Status* is transformed into a static status accompanied by a sequence of availability status transactions. The static status, encapsulated by the set $\{g_k, p_k, N_k\}$, is recorded on the blockchain a single time. Subsequently, when the charging service provider observes the variables $\boldsymbol{c}_k(t)$ and $\boldsymbol{A}_k(t)$, it initiates the function `Submit Charging Status()` with the inputs $\{\boldsymbol{c}_k(t), \boldsymbol{A}_k(t)\}$. Following this, the updated information is preserved within the local storage of the miners.

`Submit Charging Request()`: In *Definition* 2, we articulate the composition of a charging request. Consequently, when an electric vehicle solicits a charging service, it may execute the function `Submit Charging Request()` with the input parameters $\{i, g_j, E_j, s_j, \overline{d}_j\}$. Notably, the parameter a_j is not supplied by the EV but is instead documented by the smart contract.

`Scheduling()`: The main objective of this function is to solve ***oum*** based on the collected information extracted from the storage. However, ***oum*** formulates the offline problem assuming knowledge of all suppliers in the system and relevant information about charging requests, a condition unattainable in reality. Additionally, ***oum*** itself presents complexities as a linear programming problem with binary variables. To address these challenges comprehensively, we transform this problem into an online knapsack problem [29]. Each charging provider can be likened to a knapsack, with its number of charging piles N_k representing the knapsack's capacity. Charging requests correspond to items, with each item having a size of 1 and a value of v_j^k. However, unlike the traditional knapsack problem, ***oum*** still presents unique challenges compared to state-of-the-art approaches.

i) In traditional models of the online knapsack problem, item sizes are typically assumed to be much smaller than the knapsack capacity. However, in ***oum***, items may have sizes equal to the knapsack capacity. While the most advanced methods have extended this limit to $\varepsilon \leq C \ln 2/\gamma_0$ [30], where ε is the upper bound of the item size and C is the capacity of the knapsack. γ_0 is a algorithm related factor with $\gamma_0 > \ln 2$, they still cannot be directly applied to ***oum***.
ii) In the online knapsack problem, the knapsack capacity is typically assumed to be fixed. However, in ***oum***, the knapsack capacity varies over time. Specifically, the capacity of each charging provider is not static as N_k, but varies dynamically as $\boldsymbol{A}_k(t)$. This dynamic nature of the knapsack capacity adds additional complexity to the problem solution.

To effectively address the aforementioned challenges, we modify the threshold given in [30] to generate a threshold-based online algorithm. We postulate that each charging ser-

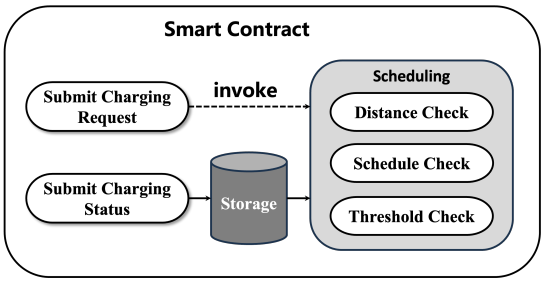


Fig. 3: Smart Contract. The main functions contained in the smart contract include `Submit Charging Status()`, `Submit Charging Request()` and `Scheduling()`.

vice provider represents a virtual knapsack, with a capacity denoted as shown in (7).

$$C_k = \max\{N_k, \frac{\gamma}{\ln 2}\}. \tag{7}$$

The capacity C_k satisfies $C_k \geq N_k$, implying that a charging request within the N_k capacity of the original charging service provider can also be accommodated by a knapsack with a capacity of C_k. Furthermore, γ is defined under two foundational assumptions, *Assumptions* 1-2 and Lemma 3, which will be explained further in section 5.1.

Assumption 1 (Limited Time-Averaged Value)**.** *The time-averaged utility of a charging request for a charging provider, represented as* $\rho_j^k = \frac{v_j^k}{d_j^k}$*, is bounded, i.e.,* $\forall j$, $\rho_j^k \in [1, \overline{V}_k]$.

Assumption 2 (Limited Duration)**.** *The charging duration for charging request* j *at provider* k *is bounded, i.e.,* $\forall j,\ d_j^k \in [\underline{D}_k, \overline{D}_k]$ $(0 < \underline{D}_k \leq \overline{D}_k < +\infty)$.

The intuition behind typical online knapsack problem algorithms involves the utilization of threshold functions to control the admission of items. This enables the algorithm to reject lower-value items and reserve capacity for potential higher-value items in the future. We utilize the threshold function introduced in [30], as outlined below.

$$\phi_k(s_k(t)) = \exp\left(\frac{s_k(t)\gamma}{C_k}\right) - 1. \tag{8}$$

where $s_k(t)$ represents the real-time knapsack utilization of k in time slot t. The function $\phi_k(s_k(t))$ is regarded as the marginal revenue corresponding to knapsack utilization $s_k(t)$. Therefore, for a charging request placed in k, the total expected revenue of the service provider can be expressed as $\omega = \sum_{t \in T_j^k} \phi_k\left(s_k(t)\right)$.

Subsequently, the scheduling process can be segmented into three distinct procedures.

`Distance Check()`: The smart contract assesses the geographical coordinates of the charging service provider k and the electric vehicle j. Should the distance between them, denoted as $dis(g_k, g_j)$, fall below the threshold ζ, it returns `True`.

`Schedule Check()`: The smart contract accesses the storage to retrieve $\boldsymbol{A}_k$. If the condition $s_k(t) + 1 \leq \boldsymbol{A}_k(t)$ is satisfied for all t in the set T_j^k and $\overline{d}_j \geq d_j^k$, it returns `True`.

The charging service providers that successfully meet the criteria of both the `Distance Check` and the `Schedule Check` constitute a subset of the optional set O_j, as delineated in *Definition* 2, and also meet the constraint (4).

`Threshold Check()`: The final evaluation concerns the utility of the charging request. Specifically, if the utility of the charging request, denoted as v_j^k, exceeds ω, the function returns `True`.

The algorithm is summarized in Algorithm 2.

Algorithm 2: Scheduling Algorithm

Input: Charging request R_j
Output: $\{x_j^k\}_{\forall k \in \mathcal{SP}}$
`// Initialization Phase;`
$\hat{x}_j^k \leftarrow 0,\ \forall\ k \in \mathcal{SP}$;
`// Process Phase;`
for $k \in \mathcal{SP}$ **do**
  $\boldsymbol{A}_k(t),\ s_k(t) \leftarrow$ extract from the storage;
  if *`Distance Check()` and `Schedule Check()`* **then**
    $\omega = \sum_{t \in T_j^k} \phi_k\left(s_k(t)\right)$;
    if *`Threshold Check()`* **then**
      $\hat{x}_j^k = 1$;
    else
      $\hat{x}_j^k = 0$;
    end
  end
end
if $\sum_{k \in \mathcal{SP}} \hat{x}_j^k > 0$ **then**
  $k^* = \arg\max_{k \in \mathcal{SP}: \hat{x}_j^k = 1} v_j^k$;
  $x_j^{k^*} = 1$ and $x_j^k = 0,\ \forall k \in \mathcal{SP} \setminus \{k^*\}$;
else
  $x_j^k = 0,\ \forall\ k \in \mathcal{SP}$;
end
$s_k(t) = s_k(t) + x_j^k,\ \forall\ k \in \mathcal{SP},\ t \in T_j^k$;

Theorem 1. *The computational complexity of the scheduling algorithm is* $\mathcal{O}(\overline{D}|\mathcal{SP}|)$*, where* $\overline{D} = \max\{\overline{D}_k\}$.

Proof. The primary limitation of the algorithm is the loop, which executes at most $|\mathcal{SP}|$ times, where $|\mathcal{SP}|$ denotes the maximum number of candidate service providers. Each iteration of the loop involves the `Scheduling Check()` function, which entails a computational complexity of at most $\mathcal{O}(\overline{D})$. □

4.4 Cryptographic Components

The proposed charging system is based on the permissionless blockchain, inherently providing privacy and partial security. For users seeking enhanced security measures, we demonstrate that our system supports various pluggable cryptographic components, thereby offering improved security guarantees.

Zero-knowledge proofs (ZKP) enable charging service providers and EVs to demonstrate their ownership or request of energy without disclosing any other private information. The system defines a language L and a proof system Π. The service provider or EV generates a proof π that allows the user to verify (x, π) such that $V(x, \pi) = True$, thereby validating the authenticity of x without revealing any information about x.

Homomorphic encryption (HE) enables the system to execute computations on encrypted data without necessitating decryption. Given the secret key SK and the public key PK, the system defines an encryption function E and a decryption function D such that $D(SK, E(PK, m_1) \oplus E(PK, m_2)) = m_1 + m_2$. Consequently, malicious users are unable to recover the original data m from $E(PK, m)$.

Multi-signature (MS) allows multiple users to jointly sign a message, ensuring the integrity of the message and the authentication of its source. The system defines a signing function S. For a set of users $U = \{u_1, u_2, ..., u_n\}$ and a message m to be signed, a signature $\sigma = S(U, m)$ is generated. The verification function V ensures that $V(m, \sigma) = True$ only when all users' signatures are valid.

5 Analysis

In this section, we present an optimization-focused analysis alongside a comprehensive security analysis.

5.1 Optimization-focused Analysis

The competitive ratio is commonly used to measure the performance of an algorithm. To analyze the competitive ratio of ***oum***, we employ the proof strategy from the typical online multi-knapsack problem with departures (OKD) [30]. This approach involves simplifying the analysis of a multi-knapsack problem to that of a single-knapsack problem and then considering two possible scenarios within the scheduling process of the single-knapsack problem:

- Case I: The knapsack capacity is sufficient, and a request can only be rejected due to the `Threshold Check`.
- Case II: The knapsack capacity is limited, and a request can be rejected due to either the `Schedule Check` or the `Threshold Check`.

It is important to clarify the distinctions between ***oum*** and OKD in the context of the following three aspects:

- **Knapsack Capacity**: The knapsack capacity in ***oum*** can be larger than the actual knapsack capacity of the charging service provider. It is necessary to ensure that the virtual knapsack does not receive charging requests that exceed the actual capacity.
- **Distance Check**: The proof process for OKD does not consider a `Distance Check` step, whereas ***oum*** requires ensuring that requests not meeting the distance criteria are not accepted.
- **Time-Varying Capacity**: The knapsack capacity in ***oum*** varies over time, in contrast to the fixed capacity in OKD. This necessitates addressing the complexities introduced by the dynamic nature of the capacity, as managed by the `Submit Charging Status()` function.

To comprehensively address the aforementioned three issues, we propose the concept of a virtual request. By introducing this concept, the proof methodology for the OKD problem can be naturally adapted to the ***oum*** problem. The overall proof strategy is summarized in Fig. 4.

5.1.1 Decomposition

We first define K ancillary problems for ***oum*** with K charging service providers. Let ***oum-k*** denote the k-th ancillary problem, which only allows charging requests assigned to the k-th knapsack (charging service provider). Subsequently, the theorem borrowed from OKD serves to bridge the relationship between ***oum-k*** and ***oum***.

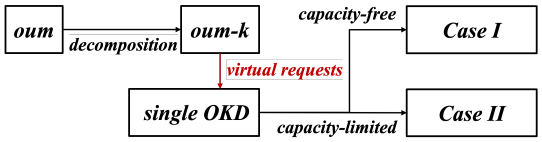


Fig. 4: Key Idea.

Theorem 2 (Problem Decomposition)**.** *Given* ***oum-k*** *is r_k-competitive for $k \in \mathcal{SP}$, then $r = 1 + \max\{r_k\}_{\forall k \in \mathcal{SP}}$, where r_k is the competitive ratio of* ***oum-k*** *and r is the competitive ratio of* ***oum****.*

5.1.2 Virtual Requests

Subsequently, our attention shifts to the simplified *oum-k*. To address the complexities arising from the differences between ***oum*** and OKD, a set of "virtual requests" is introduced to occupy the capacity.

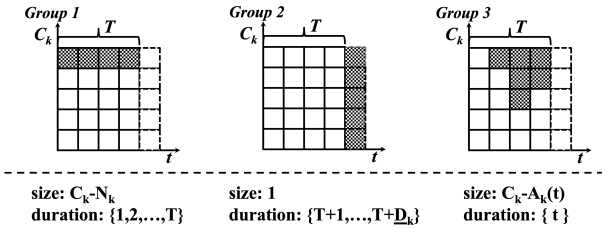


Fig. 5: Virtual Requests (The shaded areas represent virtual requests.)

As illustrated in Fig. 5, virtual requests are categorized into three types. The first type occurs when the virtual knapsack's capacity exceeds the total number of charging piles, thus the size of virtual requests is set as $C_k - N_k$. The second type pertains to the period after $T+1$, where requests failing the `distance check` are considered submitted during these already occupied times. The third type involves cases where the knapsack capacity exceeds the available charging piles $\boldsymbol{A}_k(t)$, so the size of these virtual requests is $C_k - \boldsymbol{A}_k(t)$.

Formally, the complete set of virtual requests is denoted by $\mathcal{VR} = \{1, 2, \cdots, m, \cdots, M\}$, where each request R_m^k is represented as $\{v_m^k, T_m^k\}$, and $|T_m^k| = d_m^k$. We assume they have certain values v_m^k such that they can pass the `threshold check`. To characterize them as normal requests, they should also satisfy *Assumptions* 1 and 2. Naturally, a virtual request can be divided into smaller requests to be $\leq \overline{D}_k$; however, if $d_m^k < \underline{D}_k$, this division is not feasible. Additionally, the value density of a virtual request can be larger than $\overline{V}_k$. We provide additional assumptions.

Assumption 3 (Limited Time-Averaged Virtual Value)**.** *The time-averaged value of a virtual request of a charging provider, represented as $\rho_m^k = \frac{v_m^k}{d_m^k}$, is bounded, i.e., $\forall m$, $\rho_m^k \in [1, \overline{V}_k^v]$.*

Assumption 4 (Lower Bounded Duration)**.** *The duration for virtual request m is lower bounded, i.e., $\forall m,\ d_m^k \geq \underline{D}_k^v$.*

We denote the solution given by `Scheduling` of ***oum-k*** as $ALG(CR_k, VR_k)$ and the optimal solution as $OPT(CR_k, VR_k)$ (OPT also accepts all virtual requests). Without loss of generality, we assume the timeline can be partitioned into $L = \frac{T}{\overline{D}_k}$ parts. Then the l-th interval is $T^l = \{t \in \mathcal{T} : (l-1)\overline{D}_k + 1 \leq t \leq l\overline{D}_k\}$ and I^l represents requests whose starting times are in T^l. Further, $\hat{I}^l = \hat{I}^{l-1} \cup \hat{I}^l \cup \hat{I}^{l+1}$ and $I^0 = I^{L+1} = \emptyset$. Then, the competitive ratio is given as

$$\frac{\mathrm{OPT}(CR_k, VR_k)}{\mathrm{ALG}(CR_k, VR_k)} = \frac{\sum_{l\in[L]} \mathrm{OPT}\left(I^l\right)}{\sum_{l\in[L]} \mathrm{ALG}\left(I^l\right)} \tag{9}$$

$$= \frac{3\sum_{l\in[L]} \mathrm{OPT}\left(I^l\right)}{\mathrm{ALG}\left(I^1\right) + \sum_{l\in[L]} \mathrm{ALG}\left(\hat{I}^l\right) + \mathrm{ALG}\left(I^L\right)} \tag{10}$$

$$\leq 3 \max_{l\in[L]} \frac{\mathrm{OPT}\left(I^l\right)}{\mathrm{ALG}\left(\hat{I}^l\right)} \tag{11}$$

Then, $\frac{\mathrm{OPT}(I^l)}{\mathrm{ALG}(\hat{I}^l)}$ is analyzed in two cases. To streamline the representation, we denote the set of both real and virtual requests submitted to k as $\mathcal{I}$. Within this set, the utility of $n \in \mathcal{I}$ is denoted as v_n, and the corresponding time is represented by T_n ($|T_n| = d_n$). Concurrently, we omit the subscripts in $\phi_k(s_k(t))$ and $s_k(t)$ for clarity and simplicity.

5.1.3 Case I

The final utilizations of all time slots in $\hat{T}^l = T^l \cup T^{l+1}$ are below the capacity of the virtual knapsack, i.e. $\forall\ t \in \hat{T}^l,\ s^{|\mathcal{I}|}(t) \leq C_k - 1$, which means there still exists room for a charging request. In this case, the only reason why one item is rejected by `Scheduling` is that it fails to pass the `Threshold Check`.

Theorem 3 (Lower Bound of ALG in Case I)**.** *The utility of requests in $\hat{I}^l$ is lower bounded by*

$$\mathrm{ALG}\left(\hat{I}^l\right) \geq \frac{\ln 2}{2\gamma} \sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k. \tag{12}$$

Proof. Since the initial knapsack is empty $\phi(0) = 0$, we have $\phi(s^{|\mathcal{I}|}(t)) = \sum_{n\in\mathcal{I}}[\phi(s^n(t)) - \phi(s^{n-1}(t))]$ and

$$\sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k = \sum_{t\in\hat{T}^l} \sum_{n\in\mathcal{I}} C_k\left[\phi(s^n(t)) - \phi(s^{n-1}(t))\right] \tag{13}$$

$$= \sum_{t\in\hat{T}^l} \sum_{n\in\hat{I}^l : t\in T_n} C_k\left[\phi(s^n(t)) - \phi(s^{n-1}(t))\right] \tag{14}$$

$$\leq \sum_{n\in\hat{I}^l} \sum_{t\in T_n} C_k\left[\phi(s^n(t)) - \phi(s^{n-1}(t))\right], \tag{15}$$

The second equality holds because the maximum duration of each request is $\overline{D}_k$ and hence the requests that can stay in $\hat{T}^l$ must be from $\hat{I}^l$. The last inequality holds since the requests in $\hat{I}^l$ can stay up to segment $l+2$, which is outside $\hat{T}^l$. Then we turn to analyze $\phi(s^n(t)) - \phi(s^{n-1}(t))$. As for a charging request, if it is rejected, we have $\phi(s^n(t)) = \phi(s^{n-1}(t))$, therefore $\sum_{t\in T_n} C_k\left[\phi(s^n(t)) - \phi(s^{n-1}(t))\right] = 0$. Otherwise, $\phi(s^n(t)) - \phi(s^{n-1}(t)) = 1$.

$$\sum_{t\in T_n} C_k\left[\phi(s^n(t)) - \phi(s^{n-1}(t))\right] \tag{16}$$

$$= C_k \sum_{t\in T_n} \exp\frac{s^{n-1}(t)\gamma}{C_k}\left[\exp(\frac{\gamma}{C_k}) - 1\right] \tag{17}$$

$$\leq C_k \sum_{t\in T_n} \exp(\frac{s^{n-1}(t)\gamma}{C_k}) \cdot \frac{\gamma}{C_k \ln 2} \tag{18}$$

$$= \frac{\gamma}{\ln 2} \sum_{t\in T_n} \phi\left(s^{n-1}(t)\right) + \frac{\gamma}{\ln 2} d_n \tag{19}$$

$$\leq \frac{\gamma}{\ln 2} v_n + \frac{\gamma}{\ln 2} v_n = \frac{2\gamma}{\ln 2} \Delta\mathrm{ALG}_n. \tag{20}$$

Inequality (18) holds since $\exp(x\ln 2) - 1 \leq x$ if $0 \leq x \leq 1$, and $\frac{\gamma}{C_k \ln 2} \leq 1$ based on the definition of C_k. The last inequality holds since n can pass the threshold check and the *Assumption* 1. In conclusion,

$$\sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k \leq \sum_{n\in\hat{I}^l} \frac{2\gamma}{\ln 2} \Delta\mathrm{ALG}_n = \frac{2\gamma}{\ln 2} \mathrm{ALG}\left(\hat{I}^l\right) \tag{21}$$

□

Theorem 4 (Upper Bound of OPT in Case I)**.** *The value of requests in I^l accepted by OPT is upper bounded by*

$$\mathrm{OPT}\left(I^l\right) \leq \mathrm{ALG}\left(I^l\right) + \sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k. \tag{22}$$

Proof. Using S_l to denote the set of requests in I^l accepted by ALG, and S_l^* to denote the set of requests accepted by OPT, we have $\sum_{n\in\mathcal{S}_l\cap\mathcal{S}_l^*} v_n \leq \mathrm{ALG}\left(I^l\right)$, and

$$\sum_{n\in\mathcal{S}_l^*\setminus\mathcal{S}_l} v_n \leq \sum_{n\in\mathcal{S}_l^*\setminus\mathcal{S}_l} \sum_{t\in T_n} \phi\left(s^{n-1}(t)\right) \tag{23}$$

$$\leq \sum_{n\in\mathcal{S}_l^*\setminus\mathcal{S}_l} \sum_{t\in T_n} \phi\left(s^{|\mathcal{I}|}(t)\right) \tag{24}$$

$$= \sum_{t\in\hat{T}^l} \sum_{n\in\mathcal{S}_l^*\setminus S_l : t\in T_n} \phi\left(s^{|\mathcal{I}|}(t)\right) \tag{25}$$

$$\leq \sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k. \tag{26}$$

The first inequality holds because the request fails to pass the `Threshold Check`. □

Lemma 1 (Upper Bound in Case I)**.**

$$\frac{\mathrm{OPT}\left(I^l\right)}{\mathrm{ALG}\left(\hat{I}^l\right)} \leq \frac{\mathrm{ALG}\left(I^l\right) + \sum_{t\in\hat{T}^l} \phi\left(s^{|\mathcal{I}|}(t)\right) C_k}{\mathrm{ALG}\left(\hat{I}^l\right)} \leq 1 + \frac{2}{\ln 2}\gamma \tag{27}$$

5.1.4 Case II

At least one time slot's utilization approaches the capacity of the virtual knapsack, i.e., $C_k - 1 < s^{|\mathcal{I}|}(t) \leq C_k$.

Theorem 5 (Lower Bound of ALG in Case II)**.** *The value of requests in $\hat{I}^l$ is lower bounded by*

$$\mathrm{ALG}\left(\hat{I}^l\right) \geq \frac{\ln 2 C_k \min\{\underline{D}_k, \underline{D}_k^v\}}{\gamma}[\exp(\frac{\gamma - \ln 2}{2}) - 1]. \tag{28}$$

Proof. We consider an interval with $2\min\{\underline{D}_k, \underline{D}_k^v\} - 1$ time slots, which can be considered as the intersection of two groups of requests. Assume the quantity of the first group (the number of requests) is z and the quantity of the second group is $C_k - 1 - z$, we have

$$\mathrm{ALG}\left(\hat{I}^l\right) \tag{29}$$

$$\geq \frac{\ln 2}{2\gamma}[(\min\{\underline{D}_k, \underline{D}_k^v\} - 1)\phi(z) \tag{30}$$

$$+ (\min\{\underline{D}_k, \underline{D}_k^v\} - 1)\phi(C_k - 1 - z) \tag{31}$$

$$+ \phi(C_k - 1)]C_k \tag{32}$$

$$\geq \frac{\ln 2}{2\gamma}[2(\min\{\underline{D}_k, \underline{D}_k^v\} - 1)\phi(\frac{C_k - 1}{2}) + \phi(C_k - 1)]C_k \quad (33)$$

$$\geq \frac{\ln 2 C_k \min\{\underline{D}_k, \underline{D}_k^v\}}{\gamma}[\exp(\frac{\gamma - \ln 2}{2}) - 1]. \quad (34)$$

The first inequality holds based on Theorem 3. The second inequality is obtained by taking the derivative of z to find the minimum of the original formula. The final inequality is justified by the fact that $1 \leq \frac{C_k \ln 2}{\gamma}$, with other terms omitted. □

Theorem 6 (Upper Bound of OPT in Case II)**.** *The value of requests in I^l accepted by OPT is upper bounded by*

$$\mathrm{OPT}\left(I^l\right) \leq 2C_k \max\{\overline{V}_k^v, \overline{V}_k\}\overline{D}_k. \quad (35)$$

Lemma 2 (Upper Bound in Case II)**.**

$$\frac{\mathrm{OPT}\left(I^l\right)}{\mathrm{ALG}\left(\hat{I}^l\right)} \leq \frac{2\gamma}{\ln 2} \cdot \frac{\max\{\overline{V}_k^v, \overline{V}_k\}\overline{D}_k}{\min\{\underline{D}_k, \underline{D}_k^v\}[\exp(\frac{\gamma - \ln 2}{2}) - 1]}. \quad (36)$$

Lemma 3. *The competitive ratio corresponding to $CR_k \cup VR_k$ is*

$$r \leq 3\max\left\{1 + \frac{2}{\ln 2}\gamma, \frac{2\gamma}{\ln 2} \cdot \frac{\max\{\overline{V}_k^v, \overline{V}_k\}\theta}{\exp(\frac{\gamma - \ln 2}{2}) - 1}\right\}. \quad (37)$$

where $\theta = \frac{\overline{D}_k}{\min\{\underline{D}_k, \underline{D}_k^v\}}$ and $\gamma = 2\ln(\theta\overline{V}_k + 1) + \ln 2$.

Lemma 4 (Parameterized Competitive Ratio)**.** *The order of the competitive ratio of* `Scheduling` *is related to $\overline{V}$, θ, and the portion of virtual requests.*

$$OPT \leq \mathcal{O}(\eta \ln \theta\overline{V})(ALG + B), \quad (38)$$

where $\eta = \frac{\max\{\overline{V}_k^v, \overline{V}_k\}}{\overline{V}_k}$, $\overline{V} = \max\{\overline{V}_k\}$ and B is the total utility of virtual requests.

5.2 Security Analysis

Informal security analysis indicates that the proposed mechanism can defend against specific types of attacks.

Single Point of Failure Attack: Malicious platforms may actively go offline or refuse to provide services, attempting to disrupt the normal charging process through a single point of failure. Due to the decentralized nature of blockchain, executing such an attack would require a large number of miners to simultaneously refuse service, making the probability of this scenario negligible.

Double Spending Attack: Malicious EV owners may attempt to conduct conflicting transactions with two different charging service providers using the same asset. Since transactions must be validated by miner nodes before being recorded on the blockchain, these miner nodes will check the user's balance to determine the legitimacy of the transaction. The proposed mechanism introduces an innovative node selection algorithm that maintains the normal behavior of miners, except when there is a likelihood of 51% or more of the nodes acting maliciously.

Fairness Attack: Malicious charging service providers may obtain improper profits through means such as bribery. The proposed charging system employs a charging scheduling algorithm implemented through smart contracts, with the main scheduling process ensured by transparent and publicly accessible code logic. Charging service providers who gain unfair profits can be easily identified and penalized by miner nodes.

Privacy Attack: Malicious service providers attempt to obtain users' private information. Due to the anonymity of blockchain accounts, each user's identity is uniquely determined by a hash value. Given the one-way nature of hash functions, it is impossible to reverse-engineer a user's license plate number or owner details from the account.

6 Evaluation

Our evaluation comprises two aspects. At the algorithmic level, we utilize a real charging session dataset to implement a Python-based scenario, verifying the effectiveness of the scheduling algorithm. At the implementation level, we employ Ganache [31] to develop blockchain applications and Remix [32] to construct smart contracts, thereby assessing the deployment overhead of the proposed framework.

6.1 Experimental Settings

Data set. We use the ACN-Data [33] dataset to characterize the demand side of the charging system, which comprises over 30,000 workplace charging sessions. For the experiment, we specifically extracted 2,000 charging sessions from September 1, 2018, to October 1, 2018, spanning a total of 54 charging stations. The settings for other fixed parameters involved in the experiment are detailed in Table 3.

TABLE 3: Evaluation Settings

Parameter /Variable	Description	
	Definition	*Value*
K	#Charging Providers	54
T	#Time Slots	288
p_k	Charging Power	7 kW
J	#Charging Requests	2000
N_k	#Charging Piles	$N_k \sim U(2, 10)$
$\underline{D}^k$	duration lower bound of charging requests	2
$\underline{D}_k^v$	duration lower bound of virtual requests	2
$\overline{D}_k^v$	duration upper bound of virtual requests	288

Baselines. Regarding the scheduling algorithm, we denote the proposed mechanism as ***OUM*** and consider 6 baselines.

- Charging Pile Recommendation (CPR): We examine the Charging Pile Recommendation Algorithm in [8]. This algorithm addresses each incoming charging request contingent on the availability of charging piles. Specifically, when multiple charging piles are present, each EV opts for the pile that offers the highest utility value.
- Optimal Posted Prices for Online Cloud Resource Allocation (OPP): Scholars in [34] devise pricing functions for a foundational online resource allocation problem. We adopt their unit resource price to quantify the expense associated with accepting charging requests.
- Exponential Reservation Policy (ExpRP): The approach, introduced in [35], employs normalized utilization and a reservation function to assess admission costs. It distinguishes itself by utilizing a distinct threshold function.
- Deterministic-Selection Algorithm (Alg-D): The Algorithm D, as introduced in [36], employs a deterministic threshold to regulate the admission of charging requests.
- Randomized-Selection Algorithm (Alg-R): The Algorithm R, also introduced in [36], employs a threshold sampled from a predetermined distribution to regulate the admission of charging requests. Both Alg-R and Alg-D are designed to account for charging duration.

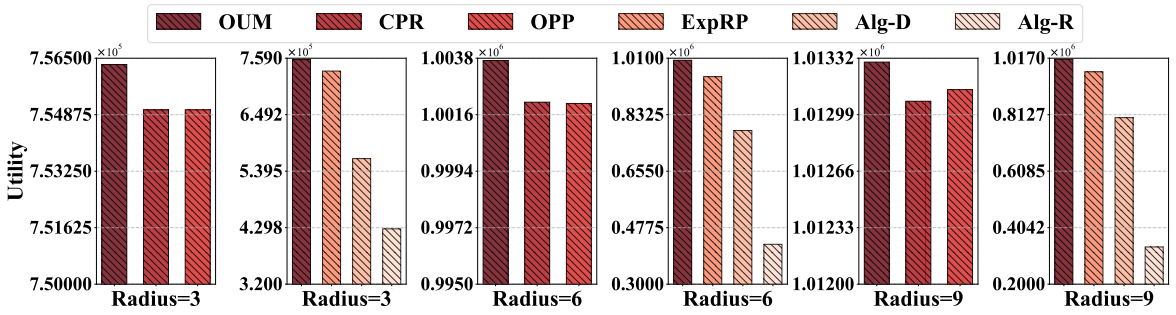


Fig. 6: Utility vs Radius

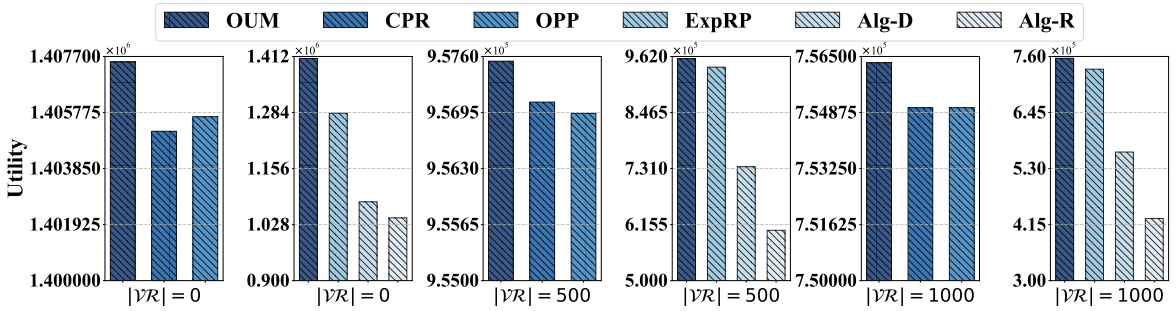


Fig. 7: Utility vs #Virtual Requests

TABLE 4: Key Parameter Settings

Setting	I	II	III	IV
$\overline{V}$	100	100	$\{80, 90, 100\}$	100
$\overline{D}$	100	100	100	$\{80, 90, 100\}$
Radius	$\{3, 6, 9\}$	3	3	3
$\lvert\mathcal{VR}\rvert$	1000	$\{0, 500, 1000\}$	1000	1000
Visualization	Fig.6	Fig.7	Table 5 (a)	Table 5 (b)

TABLE 5: Competitive Ratio ($\overline{r}$) vs Parameters

Scheduling Algorithms	(a) $\overline{V}$ 80	90	100	(b) $\overline{D}$ 80	90	100
CPR	1.670	1.670	1.670	1.668	1.669	1.670
OPP	1.670	1.670	1.670	1.668	1.669	1.670
ExpRP	1.722	2.013	1.718	1.717	1.717	1.718
Alg-D	2.236	2.236	2.236	2.163	2.182	2.236
Alg-R	1.686	1.687	1.687	1.687	1.687	1.687
OUM	**1.656**	**1.667**	**1.668**	**1.665**	**1.666**	**1.668**

In examining the competitive ratio of baseline approaches, we compute the upper bound of the optimal solution, denoted as OPT. This involves accepting all charging requests at their maximum value, as long as no conflicts arise with any virtual requests.

Metric. The primary objective of the charging system, namely the overall *Utility*, is utilized as the main metric of the experiment. Additionally, to further analyze the approximate performance of the algorithm, we use the *Competitive Ratio*, defined as $\overline{r} = \frac{OPT}{Alg}$, where Alg represents any baseline algorithm. Regarding the aspects of blockchain implementation, transaction costs, execution costs, and time consumption are taken into account.

6.2 Algorithmic Level

First, we focus on the core aspect of the proposed framework, which is the performance of the scheduling algorithm. The settings of the variable parameters corresponding to different graphs and tables are summarized in Table 4.

Fig. 6 illustrates the variation in utility relative to the number of candidate charging service providers. The term ***Radius*** denotes the optional range specified by an EV, representing the number of charging service providers that can be selected. This setting primarily indicates that, in practice, vehicles tend to choose charging stations within a limited area. Among the algorithms, CPR and OPP are more comparable to *OUM*, whereas ExpRP, Alg-D, and Alg-R exhibit poorer performance. For each configuration, two figures (with different y-axis ranges) are provided to depict the performance of these two groups of baselines relative to *OUM*. It is evident that across all radius settings, *OUM* consistently outperforms all baselines. Furthermore, as the radius increases, the utility of the methods improves. This improvement is attributed to the increase in the number of optional charging service providers, which expands the range of available resources for individual EVs, significantly enhancing optimization potential.

Fig. 7 depicts the variation in utility as a function of the number of virtual requests submitted by charging service providers. An increase in the number of virtual requests corresponds to a reduction in the available resources provided by these service providers. It is evident that across all settings, *OUM* consistently outperforms other methods. Notably, as the number of virtual requests increases, the utility achieved by all methods shows a general downward trend. This observation aligns with the intuitive understanding that a decrease in available resources leads to a corresponding reduction in utility.

To provide a more comprehensive evaluation of *OUM*'s

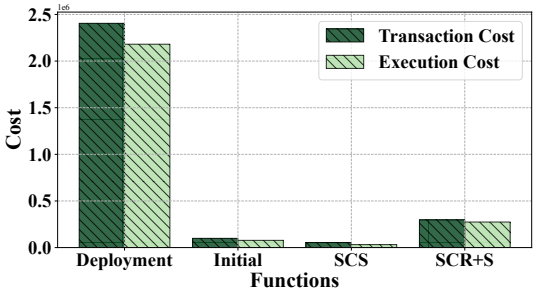


Fig. 8: Transaction and Execution Cost. (SCS represents `Submit Charging Status()` and SCR+S represents `Submit Charging Request()+Scheduling()`)

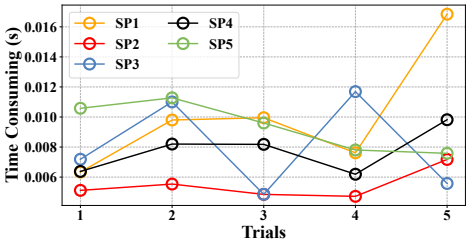


Fig. 9: Charging Service Providers' Time Consumption

TABLE 6: Environment Configuration

Attributes	Version
Operating System	macOS 14.4
Programming Language	Solidity, Python
Development Environment	Python3.12.7, Ganache v2.7.1, Remix, Web3 4.10.0

performance, we present the competitive ratios of all baselines relative to OPT as parameters $\overline{V}$ and $\overline{D}$ vary in Table 5. In each column of the table, the smallest competitive ratio is indicated in bold, while the second smallest is underlined. As depicted in the table, *OUM* consistently achieves the lowest competitive ratio across all settings. Specifically, the value-aware nature of *OUM* renders its competitive ratio more sensitive to $\overline{V}$ compared to other methods. Concerning $\overline{D}$, all methods display a certain degree of sensitivity. Given that the threshold function in the OPP method incorporates the $\overline{D}$ term, this sensitivity is to be expected. The competitive ratio of *OUM* increases with the rise in $\overline{V}$ and $\overline{D}$, corroborating the conclusion derived in Lemma 4.

6.3 Implementation Level

To evaluate our scheduling process, we implemented a prototype within Ganache, a locally deployed blockchain testing environment. Specifically, we utilized Remix to develop smart contracts encompassing all the functions depicted in Fig. 3, subsequently deploying the contract on Ganache. Following this, Python scripts were crafted to invoke Web3 functions, thereby simulating the actions of blockchain participants. The configuration details of the system environment are summarized in Table 6.

Initially, we conduct an analysis of the costs associated with deploying smart contracts and invoking key functions within the contract. The `initial()` function, in particular, represents the initialization function for each charging service provider to submit static information. For each key function, we compared the transaction costs and execution

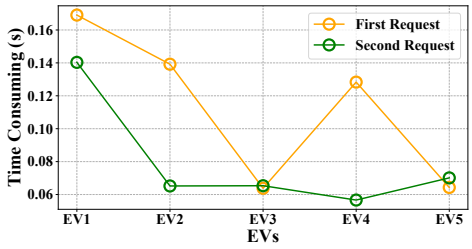


Fig. 10: EVs' Time Consumption

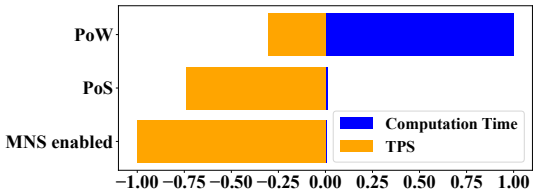


Fig. 11: Computation Cost and Throughput

costs (measured in gas). The transaction cost includes the execution cost along with additional overhead, rendering it slightly higher. We computed the average cost of five trials for each function and represented it in Fig. 8. The analysis reveals that contract deployment is the most resource-intensive process, although it generally occurs only once. Subsequently, the `Submit Charging Request()` operation, in conjunction with `Scheduling()`, emerges as the most costly recurrent operation, forming the crux of system calculations. In contrast, the costs associated with other functions remain minimal, primarily involving straightforward assignment operations.

Fig. 9 and 10 illustrate the time costs incurred by five charging service providers and five electric vehicles, respectively, when invoking the functions `Submit Charging Status()` and `Submit Charging Request()`. In Fig. 9, each of the five charging service providers (SP) submitted availability information five times. The horizontal axis represents these five trial instances, while the vertical axis displays the execution time of the respective functions. The data indicate that the time overhead for submitting charging availability information is minimal, exhibiting no discernible trend and fluctuating based on network conditions. Fig. 10 portrays the time overhead associated with five EVs, where each EV submits two charging requests. EV 1 initially submits charging requests, followed sequentially by EV 2 and subsequent EVs. Notably, since submitting a charging request involves triggering the implicit `Scheduling()` process, the time overhead is higher compared to submitting charging status. Nevertheless, the overall execution time for both functions remains relatively low.

The algorithm's complexity, as compared in Table 1, alongside the actual function execution time illustrated in Fig. 10, demonstrates the efficiency of the proposed scheduling algorithm. Furthermore, we conducted a comparative analysis of efficiency at the consensus level, as depicted in Fig. 11. The "Computation Cost" metric indicates the relative computational time overhead of various consensus methods, while "Throughput" reflects the relative transactions per second (TPS). The results reveal that the pro-

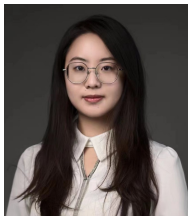
posed miner node selection (MNS) algorithm significantly conserves computing resources compared to PoW while maintaining performance comparable to PoS.

7 Conclusion

This paper has addressed gaps between supply and demand in EV charging systems by introducing a distributed blockchain-based EV charging scheduling framework, leveraging smart contracts to ensure automatic and equitable allocation of charging resources. The proposed framework is further enhanced by a novel node selection algorithm that accounts for the heterogeneous characteristics of charging service providers. Through rigorous theoretical proofs, we substantiate the guarantees of our scheduling algorithm, and our comprehensive security analysis and extensive experiments validate the framework's security and efficiency.

References

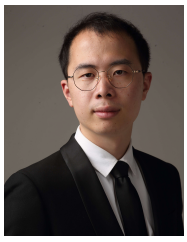
[1] IEA, "Global ev outlook 2024 - trends in electric cars," https://www.iea.org/reports/global-ev-outlook-2024.
[2] S. Bus, "Fast charging stations for electric buses installed in milan. 170 e-buses by end 2021," https://www.sustainable-bus.com/news/fast-charging-station-electric-buses-atm-milano/.
[3] Y. Wang, Z. Su, J. Li, N. Zhang, K. Zhang, K.-K. R. Choo, and Y. Liu, "Blockchain-based secure and cooperative private charging pile sharing services for vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1857–1874, 2022.
[4] X. Lu, J. Li, S. Yuan, H. Jin, C. Wu, and Z. Xu, "Toward real-time pricing and allocation for surplus resources in electric bus charging stations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 2, pp. 2101–2115, 2024.
[5] L. Liu, H. Zhang, J. Xu, and P. Wang, "Providing active charging services: An assignment strategy with profit-maximizing heat maps for idle mobile charging stations," *IEEE Transactions on Mobile Computing*, vol. 23, no. 3, pp. 2139–2152, 2024.
[6] EVBox, "Smart ev charging stations and software — evbox." [Online]. Available: https://evbox.com/en/
[7] I. ChargePoint, "Evse — electric vehicle (ev) charging stations — chargepoint." [Online]. Available: https://www.chargepoint.com/en-gb
[8] Z. Sun, P. Zhao, C. Wang, X. Zhang, and H. Cheng, "An efficient and secure trading framework for shared charging service based on multiple consortium blockchains," *IEEE Transactions on Services Computing*, vol. 16, no. 4, pp. 2437–2450, 2023.
[9] M. A. Ferrag and L. Maglaras, "Deepcoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1285–1297, 2020.
[10] Y. Li and B. Hu, "A consortium blockchain-enabled secure and privacy-preserving optimized charging and discharging trading scheme for electric vehicles," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 1968–1977, 2021.
[11] S. Lee and D.-H. Choi, "Three-stage deep reinforcement learning for privacy-and safety-aware smart electric vehicle charging station scheduling and volt/var control," *IEEE Internet of Things Journal*, vol. 11, no. 5, pp. 8578–8589, 2024.
[12] M. B. Hossain, S. R. Pokhrel, and H. L. Vu, "Efficient and private scheduling of wireless electric vehicles charging using reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 4089–4102, 2023.
[13] Y. Zhang, P. You, and L. Cai, "Optimal charging scheduling by pricing for ev charging station with dual charging modes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3386–3396, 2018.
[14] S. Wang, S. Bi, and Y. A. Zhang, "Reinforcement learning for real-time pricing and scheduling control in ev charging stations," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 849–859, 2021.
[15] Y. Ding, F. Chen, J. Feng, and H. Cheng, "Competition and pricing strategy of electric vehicle charging services considering mobile charging," *IEEE Access*, vol. 12, pp. 88 739–88 755, 2024.
[16] Y. Chen, S. Hu, S. Xie, Y. Zheng, Q. Hu, and Q. Yang, "Optimal dynamic pricing of fast charging stations considering bounded rationality of users and market regulation," *IEEE Transactions on Smart Grid*, vol. 15, no. 4, pp. 3950–3965, 2024.
[17] Z. Zhao and C. K. M. Lee, "Dynamic pricing for ev charging stations: A deep reinforcement learning approach," *IEEE Transactions on Transportation Electrification*, vol. 8, no. 2, pp. 2456–2468, 2022.
[18] L. Liu, S. Liu, J. Wu, and J. Xu, "A placement strategy for idle mobile charging stations in ioev: From the view of charging demand force," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 5, pp. 3870–3884, 2024.
[19] S. Deb, K. Tammi, X.-Z. Gao, K. Kalita, P. Mahanta, and S. Cross, "A robust two-stage planning model for the charging station placement problem considering road traffic uncertainty," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6571–6585, 2022.
[20] Y. Cao, Y. Wang, D. Li, and X. Chen, "Joint routing and wireless charging scheduling for electric vehicles with shuttle services," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 810–14 819, 2023.
[21] W. Zhang, H. Liu, F. Wang, T. Xu, H. Xin, D. Dou, and H. Xiong, "Intelligent electric vehicle charging recommendation based on multi-agent reinforcement learning," in *Proceedings of the Web Conference 2021*, 2021, pp. 1856–1867.
[22] G. Wang, X. Xie, F. Zhang, Y. Liu, and D. Zhang, "bcharge: Data-driven real-time charging scheduling for large-scale electric bus fleets," in *2018 IEEE Real-Time Systems Symposium (RTSS)*, 2018, pp. 45–55.
[23] W. Xu, W. Liang, X. Jia, H. Kan, Y. Xu, and X. Zhang, "Minimizing the maximum charging delay of multiple mobile chargers under the multi-node energy charging scheme," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1846–1861, 2021.
[24] R. Kakkar, S. Chaudhary, R. Gupta, S. Tanwar, and S. Kumar, "Blockchain-based decentralized application for electric vehicle scheduling at charging station," in *Proceedings of the 2024 ACM Southeast Conference*, ser. ACM SE '24, 2024, p. 2–8.
[25] Y. Wu, C. Zhang, and L. Zhu, "Privacy-preserving and traceable blockchain-based charging payment scheme for electric vehicles," *IEEE Internet of Things Journal*, vol. 10, no. 24, pp. 21 254–21 265, 2023.
[26] A. Jindal, G. S. Aujla, N. Kumar, and M. Villari, "Guardian: Blockchain-based secure demand response management in smart grid system," *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 613–624, 2020.
[27] D. Li, R. Chen, Q. Wan, Z. Guan, S. Li, M. Xie, J. Su, and J. Liu, "Intelligent and fair iov charging service based on blockchain with cross-area consensus," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 15 984–15 994, 2023.
[28] L. Liu, H. Zhang, J. Xu, and P. Wang, "Providing active charging services: An assignment strategy with profit-maximizing heat maps for idle mobile charging stations," *IEEE Transactions on Mobile Computing*, vol. 23, no. 3, pp. 2139–2152, 2024.
[29] C. Thielen, M. Tiedemann, and S. Westphal, "The online knapsack problem with incremental capacity," *Mathematical Methods of Operations Research*, vol. 83, pp. 207–242, 2016.
[30] B. Sun, L. Yang, M. Hajiesmaili, A. Wierman, J. C. Lui, D. Towsley, and D. H. Tsang, "The online knapsack problem with departures," in *Abstract Proceedings of the 2023 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '23, 2023, p. 59–60.
[31] T. SUITE, "Ganache - one click blockchain," https://archive.trufflesuite.com/ganache/.
[32] R. Project, "Remix - ethereum ide," https://remix.ethereum.org.
[33] Z. J. Lee, T. Li, and S. H. Low, "Acn-data: Analysis and applications of an open ev charging dataset," in *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, ser. e-Energy '19, 2019, p. 139–149.
[34] Z. Zhang, Z. Li, and C. Wu, "Optimal posted prices for online cloud resource allocation," in *Proceedings of the 2017 ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '17 Abstracts, 2017, p. 60.
[35] L. Yang, A. Zeynali, M. H. Hajiesmaili, R. K. Sitaraman, and D. Towsley, "Competitive algorithms for online multidimensional knapsack problems," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 5, no. 3, dec 2021.
[36] S. Goyal and D. Gupta, "The online reservation problem," *Algorithms*, vol. 13, no. 10, p. 241, 2020.



Xinyu Lu (Graduate Student Member, IEEE) received the B.E. degree in Computer Science (CS) from Shanghai Jiao Tong University, Shanghai, China, in 2020. She is currently pursuing the Ph.D. degree in Computer Science with Shanghai Jiao Tong University (SJTU), Shanghai, China, and working in the field of big data, cyber-physical systems (CPS), and smart transportation.

Zihan Wang (Graduate Student Member, IEEE) received the B.E. degree in Artificial Intelligence (AI) from Xi'an Jiao Tong University, Shaanxi, China, in 2023. He is currently pursuing the Ph.D. degree in Electronic and Information Engineering with Shanghai Jiao Tong University (SJTU), Shanghai, China, and working in the field of big data, blockchain, and network security.

Zhuo Chen (Student Member, IEEE) is currently pursuing A.B. degree in Computer Science from Bryn Mawr College, Pennsylvania, USA and double majoring in Pure Mathematics in Haverford College, Pennsylvania, USA. Her research interests include debugging and fault localization, software testing, Neural Networks, and smart transportation.



Jiong Lou (Member, IEEE) received the B.S. degree and Ph.D. degree in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China, in 2016 and 2023. Since 2023, he has held the position of research assistant professor in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His current research interests include edge computing, task scheduling, and container management.

Chentao Wu (Member, IEEE) received the BS, ME and PhD degrees in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2004, 2006, and 2010, respectively, and another PhD degree in electrical and computer engineering from Virginia Commonwealth University, Richmond, in 2012. He is currently a professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include computer architecture and data storage systems, and he has published more than 100 papers in prestigious international conferences and journals, such as the TC, TPDS, HPCA, DAC, IPDPS, etc. He is a member of the IEEE and ACM, a senior member of China Computer Federation.

Yuan Luo (Member, IEEE) received the B.S. degree in applied mathematics and the M.S. and Ph.D. degrees in probability statistics from Nankai University, Tianjin, China, in 1993, 1996, and 1999, respectively. From July 1999 to April 2001, he held a post-doctoral position with the Institute of Systems Science, Chinese Academy of Sciences, Beijing, China. From May 2001 to April 2003, he held another post-doctoral position with the Institute for Experimental Mathematics, University of Duisburg-Essen, Essen, Germany. Since June 2003, he has been with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. Since 2006, he has been a Full Professor and the Vice Dean of the department from 2016 to 2018 and since 2021, respectively. His current research interests include coding theory, information theory, and big data analysis.

Guangtao Xue (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2004. He is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His research interests include vehicular ad hoc networks, wireless networks, mobile computing, and distributed computing. Prof. Xue is a member of the IEEE Communication Society.

Yusheng Ji (Fellow, IEEE) received the B.E., M.E., and D.E. degrees in Electrical Engineering from the University of Tokyo. She joined the National Center for Science Information Systems (NACSIS), Tokyo, Japan in 1990. She is currently a Professor at the National Institute of Informatics (NII), Tokyo, Japan, and the Graduate University for Advanced Studies, SOKENDAI, Japan. Her research interests include network resource management and mobile computing. She is an Associate Editor of IEEE Vehicular Technology Magazine, served as Editor of IEEE Transactions of Vehicular Technology, TPC Co-chair of INFOCOM 2023, General Co-chair of ICT-DM 2018, MSN2020, BigCom2023, Symposium Co-chair of IEEE GLOBECOM 2012, 2014, and ICC 2020, and Track Co-chair of IEEE VTC 2016 Fall and 2017 Fall. She is a Distinguished Lecturer of IEEE Vehicular Technology Society.

Wei Zhao (Fellow, IEEE) received the degree in physics from Shaanxi Normal University, China, in 1977, and the M.Sc. and Ph.D. degrees in computer and information sciences from the University of Massachusetts at Amherst in 1983 and 1986, respectively. He has served in important leadership roles in academics, including the Chief Research Officer with the American University of Sharjah; the Chair of the Academic Council; CAS Shenzhen Institute of Advanced Technology; the eighth rector of the University of Macau; the Dean of Science with the Rensselaer Polytechnic Institute; the Director for the Division of Computer and Network Systems, U.S. National Science Foundation; and the Senior Associate Vice President for Research at Texas A&M University. He has made significant contributions to cyber-physical systems, distributed computing, real-time systems, and computer networks. He led the effort to define the research agenda and to create the very first funding program for cyber-physical systems in 2006. His research results have been adopted in the standard of survivable adaptable fiber optic embedded networks. He was awarded the Lifelong Achievement Award by the Chinese Association of Science and Technology in 2005.

Jie Li (Fellow, IEEE) received the B.E. degree in computer science from Zhejiang University, Hangzhou, China, the M.E. degree in electronic engineering and communication systems from China Academy of Posts and Telecommunications, Beijing, China. He received the Dr. Eng. degree from the University of Electro-Communications, Tokyo, Japan. He is with Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China where he is a chair professor. His current research interests are in big data and AI, blockchain, edge computing, networking, and security, OS, information system architecture. He serves as the director of Shanhai Jiao Tong University Blockchain Research Centre. He was a professor in Department of Computer Science, University of Tsukuba, Japan. He was a visiting Professor in Yale University, USA, Inria Sophia Antipolis and Inria Grenoble-Rhone-Aples, France. He is the co-chair of IEEE Technical Community on Big Data and the founding Chair of IEEE ComSoc Technical Committee on Big Data and the Co-Chair of IEEE Big Data Community. He has served as an associated editor for many IEEE journals and transactions. He has also served on the program committees for several international conferences.